Karlsruhe Institute of Technology

KASTEL

(Understanding) Security Trade-Offs in Cloud Storage Systems

Steffen Müller

KOMPETENZZENTRUM FÜR ANGEWANDTE SICHERHEITSTECHNOLOGIE

Source: http://www.free-picture.net/albums/nature/clouds/celestial-castel.jpg

Agenda

- Basics of Cloud Storage Systems
- A Reference Threat Model for Cloud Storage Services
- (Security) Trade-Offs in Cloud Storage Systems
  - Encryption of Data-at-Rest
  - Secure Communication
- Summary

BASICS OF CLOUD STORAGE SYSTEMS

What are Cloud Storage Systems (CSS)?

- Simple Storage Service (S3)
- DynamoDB
- …

- DocumentDB
- Redis Cache
- …

- Cloud Storage
- Data Store
- …

Cloud Storage Services

Project Voldemort

NoSQL Systems

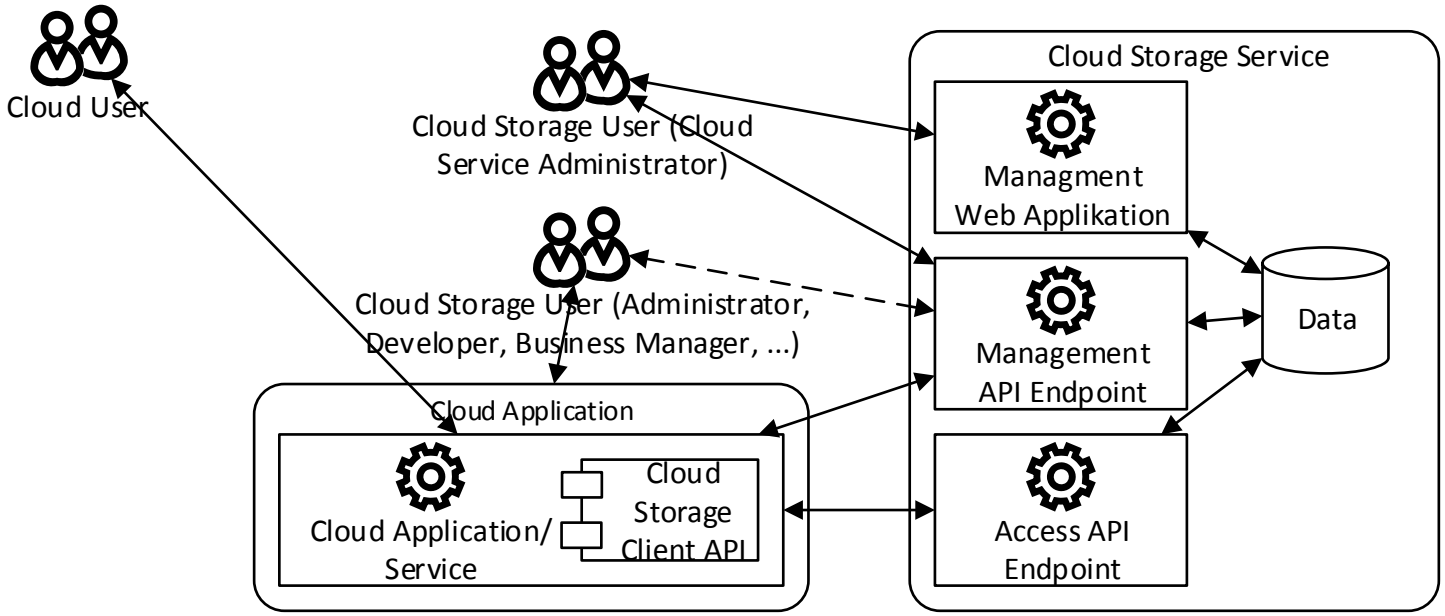
Properties of CSS

CSS are optimized for performance, availability, and elastic scalability [1]:

- Inherent trade-offs, e.g., consistency trade-offs, are typically decided in favor of performance and availability (see also: CAP Theorem [2])
- Often designed for specific use cases:
  - Amazon built Dynamo for their shopping cart purposes
  - Google designed CSS for their extraordinary workloads
  - LinkedIn invented/uses Project Voldemort for their “Who’s Viewed My Profile” functionality
  - …
- Limited query functionality compared to SQL in relational database management systems (DBMS):
  - Key-Value data model (Key-Value Store): Google Cloud Storage, S3, Redis, Voldemort, …
  - Document-oriented data model (Document Store): MongoDB, CouchDB, …
  - Column-oriented date model (Column Store): Cassandra, DynamoDB, HBase, Google Cloud Datastore, …
  - …

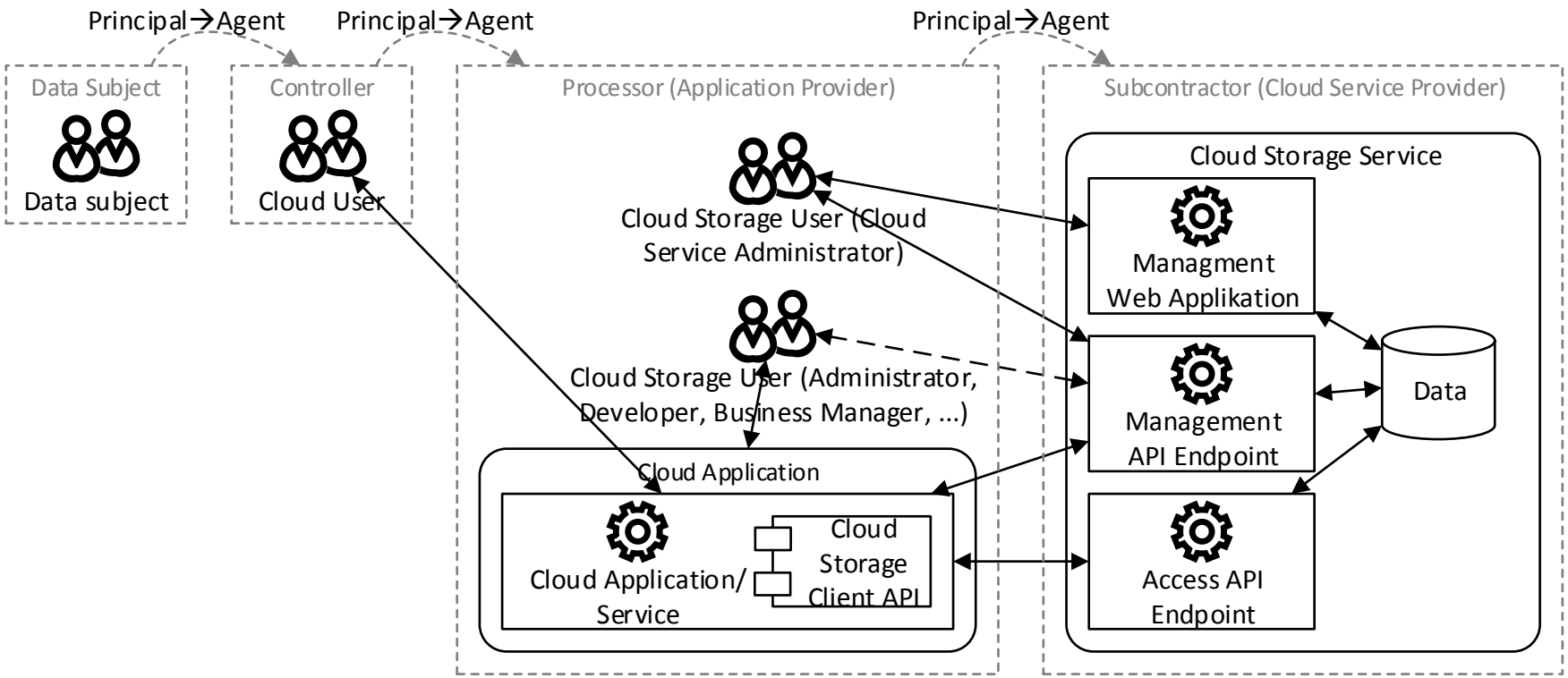
A REFERENCE THREAT MODEL FOR CLOUD STORAGE SERVICES

“Usage Model” for Cloud Storage Services



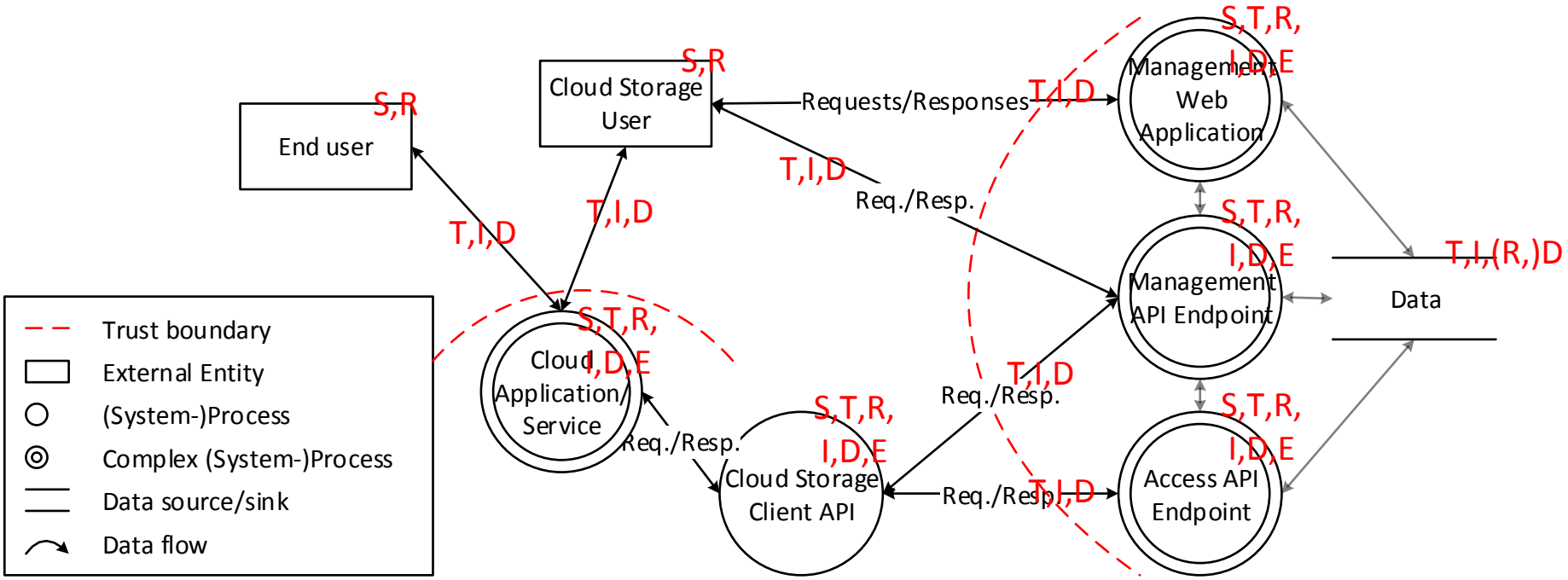
Source: [3]

“Usage Model” for Cloud Storage Services



Source: [3]

A Reference Threat Model for Cloud Storage Services

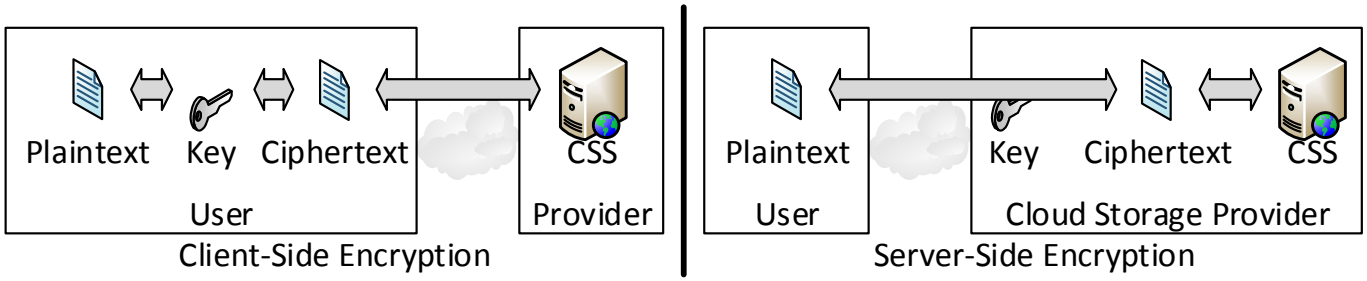


Source: [3]; see also: [4]

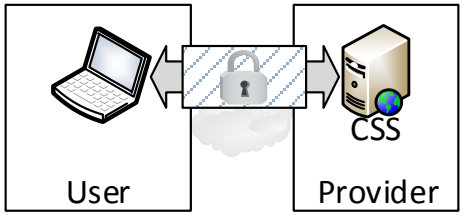
(SECURITY) TRADE-OFFS IN CLOUD STORAGE SYSTEMS

Security Mechanisms of CSS

- Most important assets are the system itself and the stored data
- General security mechanisms basically known from Security Engineering for DBMS (see also: [5, 6]), e.g.:
  - Authentication and access control
  - Encryption of data-at-rest

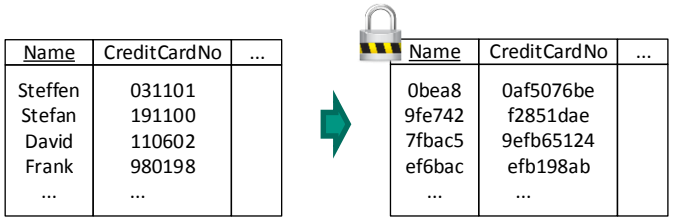


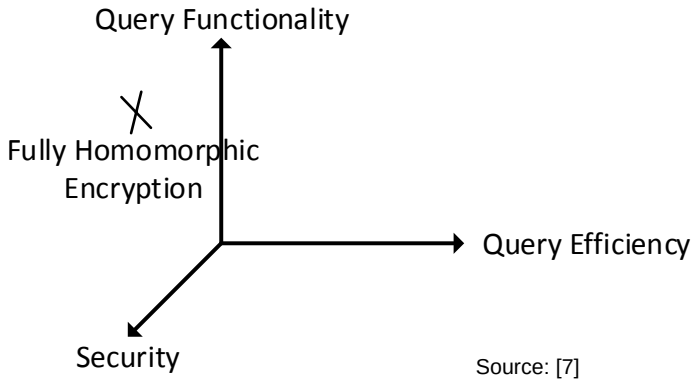
  - Secure communication (data-in-transit)



Encryption of Data-at-Rest (1/2)

- Encryption of data-at-rest aims at preserving data confidentiality (and integrity)
- Main problem [7, 8]: How do we implement querying over the encrypted data (security vs. performance)?



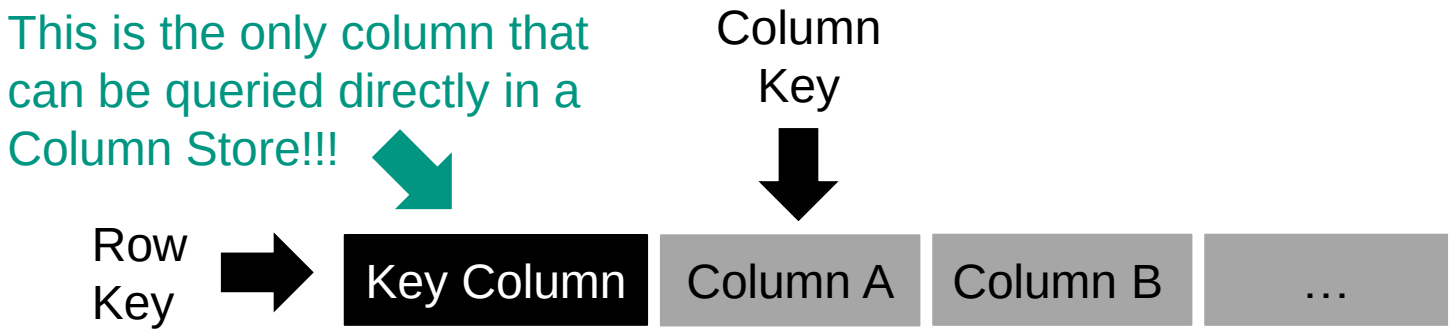


Source: [7]

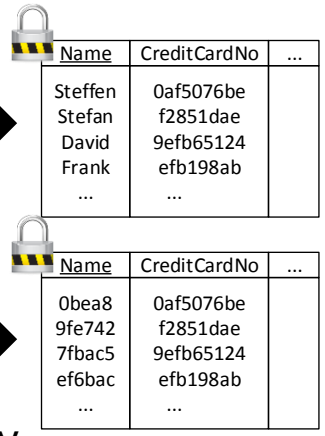
- Diverse encryption schemes as well as sophisticated architectures for CSS trying to solve the trade-off issues for CSS, e.g. [7, 8]:
  - Fully Homomorphic Encryption [9]
  - Searchable Encryption [10]
  - Cryptographic Cloud Storage [11]
  - Securus [8]
  - …

Encryption of Data-at-Rest (2/2)

- However, is encrypting data-at-rest in a CSS with its limited query functionality really a problem??? (Example: Column Store – a hashtable of hashtables; see also [12])

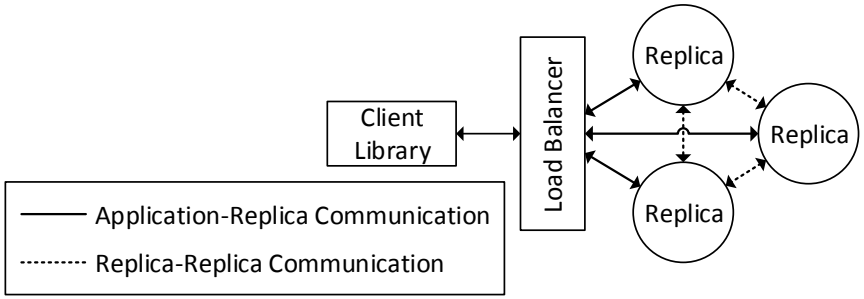


- So, it depends on the required query functionality:
  - If we do not have to encrypt a column that can be queried by the clients: **No**
  - If we have to encrypt one of the few columns which can be queried by the clients: **Maybe. But, see also**: The *Confidentiality Preserving Indexing (CPI) Taxonomy* and *Securus* Approach by Köhler in [8] allow for creating an optimal solution (query efficiency) for relational DBMS that fits the required query functionality



Secure Communication (1/3)

- Two basic types of communication in CSS [13]:
  - Application-replica (AR) communication: Comprises the data flow from the application to the first replica server including the hop via the load balancer
  - Replica-replica (RR) communication: Happens between replica servers

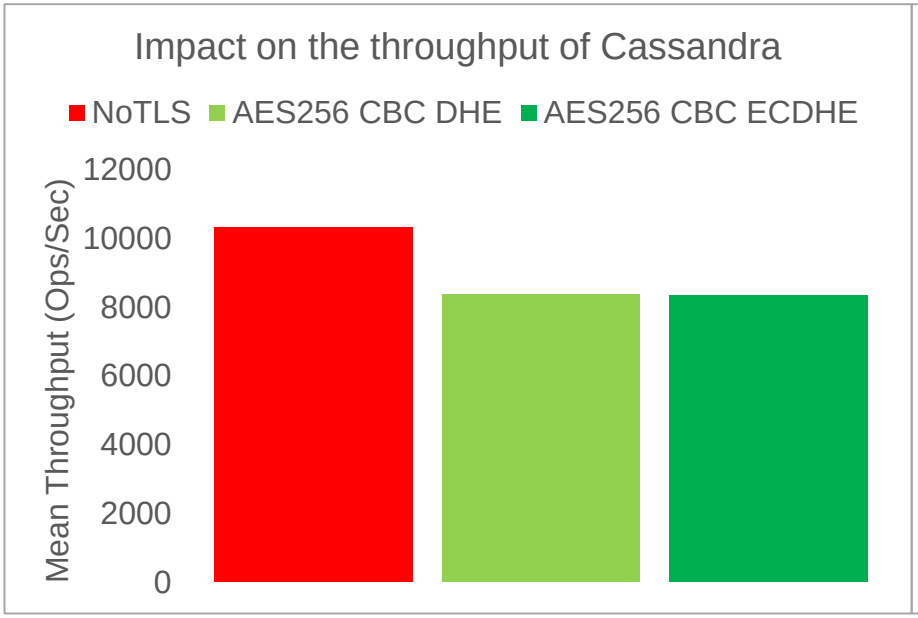


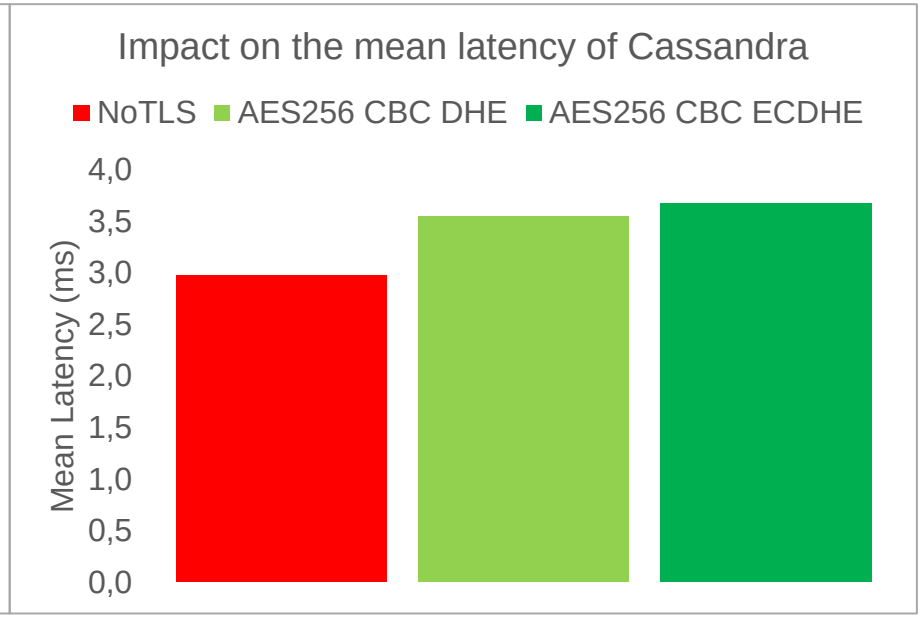
- Both communication types can be secured by, e.g., Transport Layer Security (TLS) like in MongoDB or Cassandra

- TLS officially supports more than 300 different cipher suites
- Some cipher suites are faster, some are considered to be more secure
- Enabling TLS typically reduces the throughput and heightens the latency of the communication link (security vs. performance)

Secure Communication (2/3)

Experiment DHE vs. ECDHE (AR Communication)

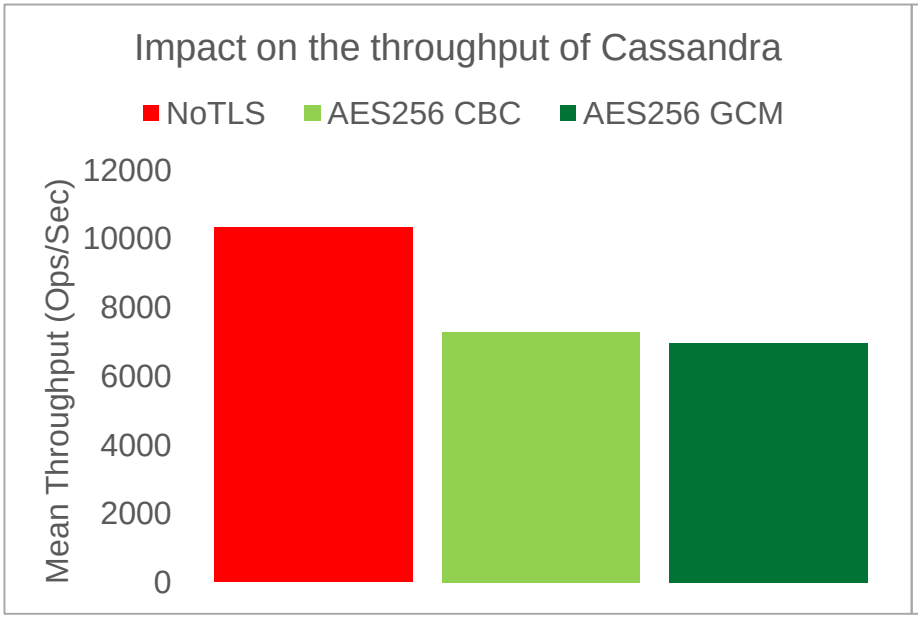


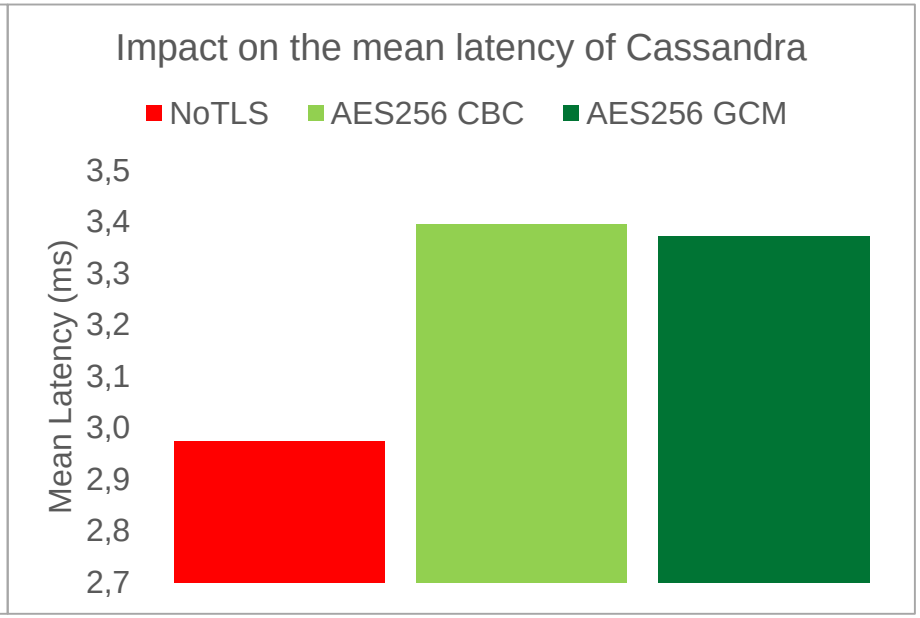


- Enabling TLS reduces the throughput by 19% in average (AR comm. secured)
- Update and read latencies are increased by 15-24% in average with TLS
- However, no significant difference between DHE and ECDHE, despite ECDHE typically outperforms DHE in web servers
- The Cassandra cluster performed only 8 (DHE)/12 (ECDHE) abbreviated handshakes and only 3/3 full handshakes during ca. 27 Min experiment runtime

Secure Communication (3/3)

Experiment CBC vs. GCM (AR-RR Communication)





- CBC mode is typically faster than the GCM, whereas the GCM is considered to be more secure – CBC have been vulnerable to different attacks like POODLE
- If we, however, activate TLS for the AR as well as RR communication, both cipher suites are nearly equal
- The throughput is reduced by 29% (CBC)/32% (GCM) and the latencies are increased by 4/6% (update) respectively 23/23% (read) in average

SUMMARY

Summary & Outlook

- Security Engineering for CSS is getting more and more important
- For Security Engineering of CSS, we have to understand the trade-offs between security and performance in CSS to make “good” decisions on the trade-offs

- In this talk, we presented:
  - A short introduction to CSS with their overall properties
  - A generic “usage model” and reference threat model for cloud storage services which can help us to better understand the threats to cloud storage services; a reference threat model for NoSQL systems is in preparation
  - A comprehensive insight into the trade-offs between security and performance for the data-at-rest encryption in CSS
  - Another comprehensive insight into some trade-offs between security and performance for secure communication

Thanks for your attention.

Steffen Müller
st.mueller@kit.edu

References

[1] P. J. Sadalage, M. Fowler: “NoSQL distilled – A brief guide to the emerging world of polyglot persistence,” Addison-Wesley, 2013.

[2] E. A. Brewer, “Towards robust distributed systems,” in PODC 2000 Keynote, 2000.

[3] S. Müller, F. Pallas, S. Balaban: “On the Security of Public Cloud Storage,” Proc. of the Future Security 2015, 2015.

[4] A. Shostack: “Threat Modeling: Designing for Security,” John Wiley & Sons, 2014.

[5] E. Bertino, S. Jajodia, P. Samarati: “Database security – Research and practice,” Information Systems, 1995.

[6] E. Bertino, R. Sandhu: “Database security – Concepts, approaches, and challenges,” Dependable and Secure Computing, IEEE Transactions on, 2005.

[7] K. Smith, M. Allen, H. Lan, A. Sillers: “Making Query Execution Over Encrypted Data Practical,” Secure Cloud Computing, Springer, 2014.

[8] J. Köhler: “Tunable Security for Deployable Data Outsourcing,” PhD Thesis, KIT Department of Informatics, 2015.

[9] C. Gentry: “Fully Homomorphic Encryption Using Ideal Lattices,” Proc. of the ACM Symposium on Theory of Computing, 2009.

[10] D. X. Song, D.Wagner, A. Perrig: “Practical techniques for searches on encrypted data,” Proc. of the IEEE Symposium on Security and Privacy, 2000.

[11] S. Kamara, K. Lauter: “Cryptographic Cloud Storage,” Financial Cryptography and Data Security, Springer, 2010.

[12] D. Bermbach, S. Müller, J. Eberhardt, S. Tai: “Informed Schema Design for Column Store-based Database Services,” Proc. of the SOCA 2015, 2015.

[13] S. Müller, D. Bermbach, F. Pallas, S. Tai: “Benchmarking the Performance Impact of Transport Layer Security in Cloud Database Systems,” Proc. of the IC2E 2014, 2014.